R. ARUNACHALAM 1-18-4-4-1

1/14

FIG. 1

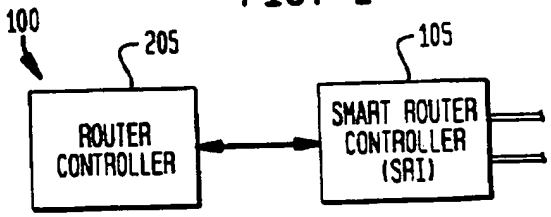


FIG. 2

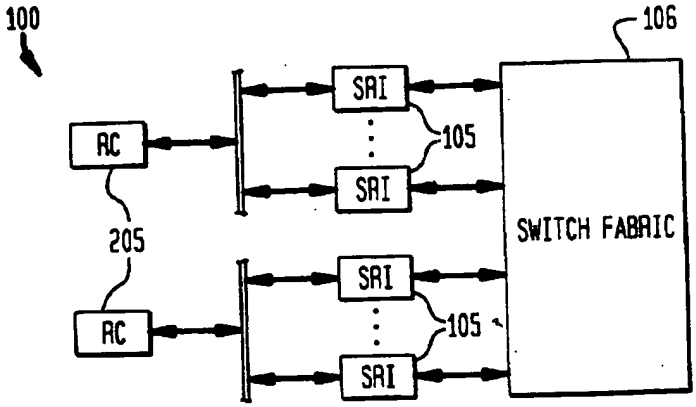
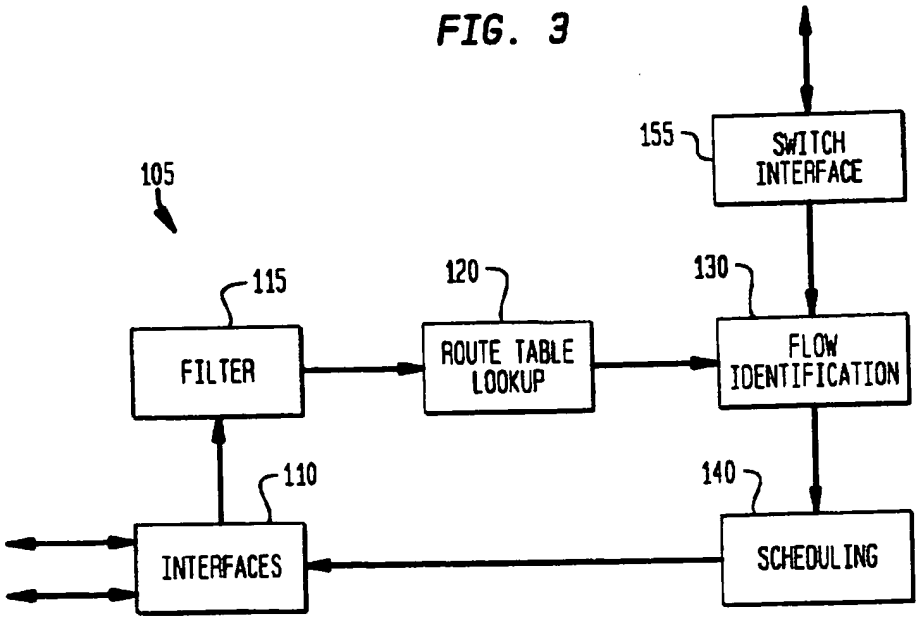


FIG. 3



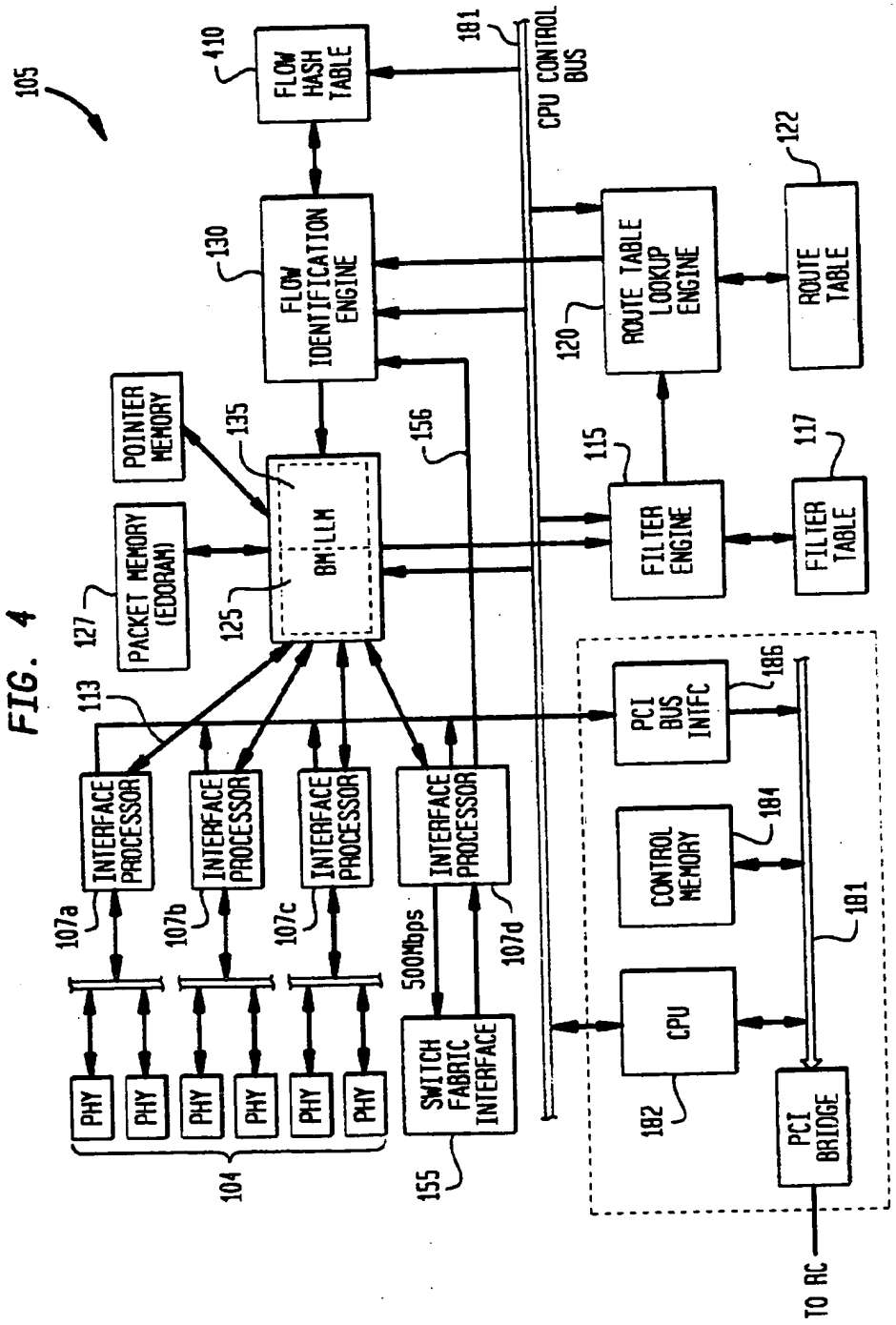
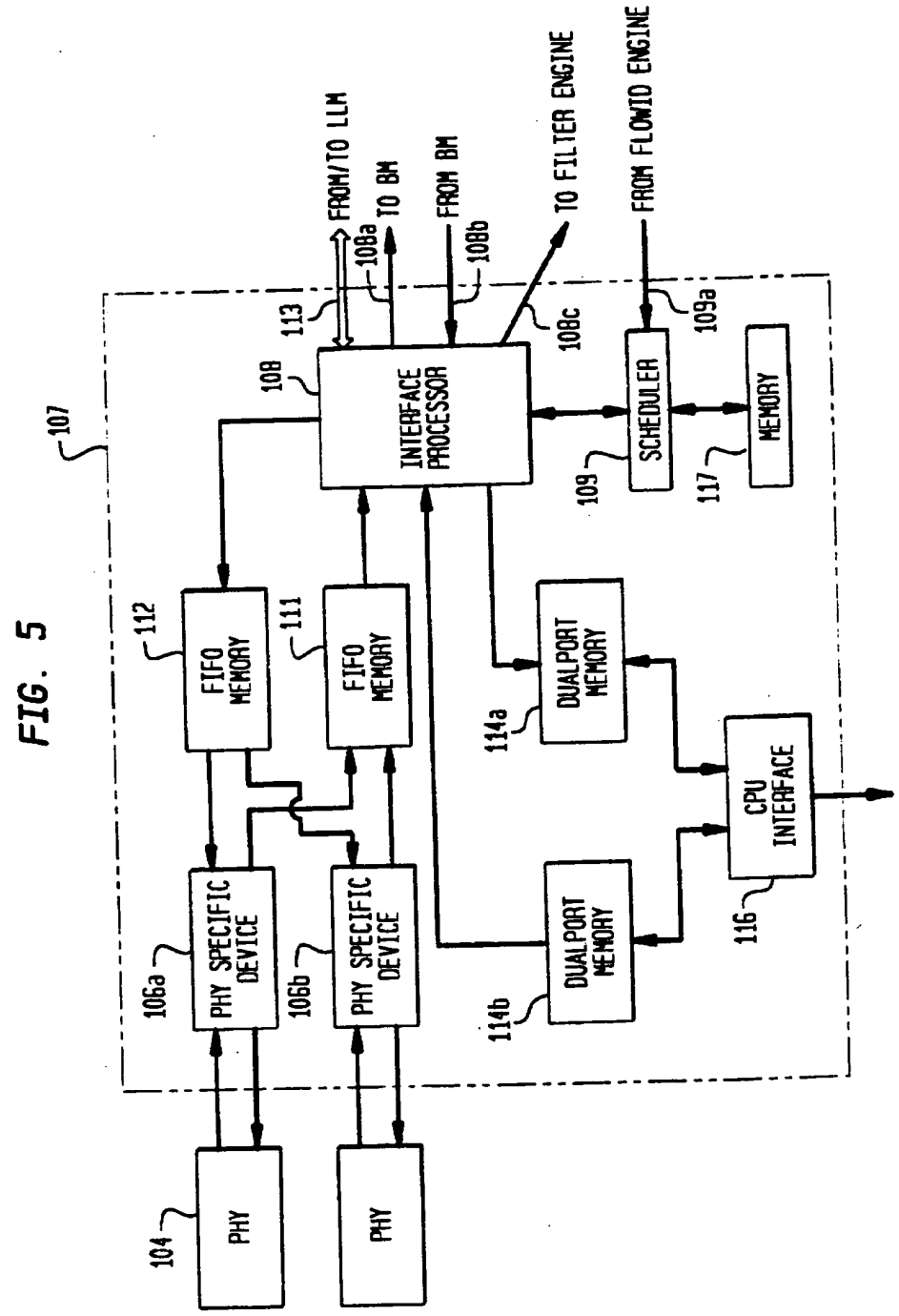

105
FLOW HASH TABLE
410
181
CPU CONTROL BUS
122
ROUTE TABLE
130
FLOW IDENTIFICATION ENGINE
ROUTE TABLE LOOKUP ENGINE
120
POINTER MEMORY
135
156
115
117
FILTER ENGINE
FILTER TABLE
PACKET MEMORY (EDORAM)
127
125
BM/LLM
113
PCI BUS INTFC
186
INTERFACE PROCESSOR
INTERFACE PROCESSOR
INTERFACE PROCESSOR
INTERFACE PROCESSOR
107a
107b
107c
107d
500Mbps
CONTROL MEMORY
184
181
CPU
182
PCI BRIDGE
PHY
PHY
PHY
PHY
PHY
PHY
104
SWITCH FABRIC INTERFACE
155
TO RC

FIG. 4

R. ARUNACHALAM 1-18-4-4-1

3/14

FIG. 5



R. ARUNACHALAM 1-18-4-4-1

4/14

FIG. 6A

209

	ADDRESS	HighByte	LowByte	
206	1	DestinationAddress(31:24)	DestinationAddress(24:16)	
	2	DestinationAddress(15:8)	DestinationAddress(7:0)	
203	3	SourceAddress(31:24)	SourceAddress(23:16)	
	4	SourceAddress(15:8)	SourceAddress(7:0)	
211	5	Destination Port(15:8)	Destination Port(7:0)	
209	6	SourcePort(15:8)	SourcePort(7:0)	
213	7	Protocol(7:0)	TimeToLive(7:0)	216
219	8	PHY Index(7:0)	PollerIndex(7:0)	221
223	9	TypeOfService(7:0)	FrameType(7:0)	226
229	10	PacketLength(15:8)	PacketLength(7:0)	
	11	TunnelSource(31:24)	TunnelSource(23:16)	231
	12	TunnelSource(15:8)	TunnelSource(7:0)	
233	13	PageListHead(31:24)	PageListHead(23:16)	
	14	PageListHead(15:8)	PageListHead(7:0)	
236	15	ControlFlags(31:24)	ControlFlags(23:16)	
	16	ControlFlags(15:8)	ControlFlags(7:0)	
239	17	RecordRoute Option Offset (7:0)	Strict Source Option Offset/ICMP Type(7:0)	241
243	18	LooseSource Option Offset/ICMP Code(7:0)	QoS Parameters(23:16)	
251	19	QoSParameters(15:8)	QoSParameters(7:0)	
249	20	FlowQueueIndex(15:8)	FlowQueueIndex(7:0)	
253	21	QoSClass(15:8)	QoSClass(7:0)	
	22	MACAddress(47:40)	MACAddress(39:32)	
246	23	MACAddress(31:24)/MulticastBitmap(31:24)	MACAddress(23:16/MulticastBitmap(23:16)	
	24	MACAddress(15:8)/MulticastBitmap(15:8)	MACAddress(7:0)/MulticastBitmap(7:0)	

FIG. 6B

236

FLAG	BIT POSITION
IcmpError	1
237 LsrrIpOption	2
239 SsrrIpOption	3
RrIpOption	4
TimestampOption	5
DontFragment	6
EndSourceRoute	7
TunnelInPacket	8
RedirectPacket	18
MacInArpCache	19
235 ClassifyQueue	20
BandWidthSpecified	21
QueueSpecified	22
UpdateArpEntry	23
NeedArpPacket	24
PacketFromRc	25
IpInIpPacket	26
TunneledPacket	27
BroadcastPacket	28
MulticastPacket	29
RcPacket	30
DropPacket	31
PassPacket	32

FIG. 7

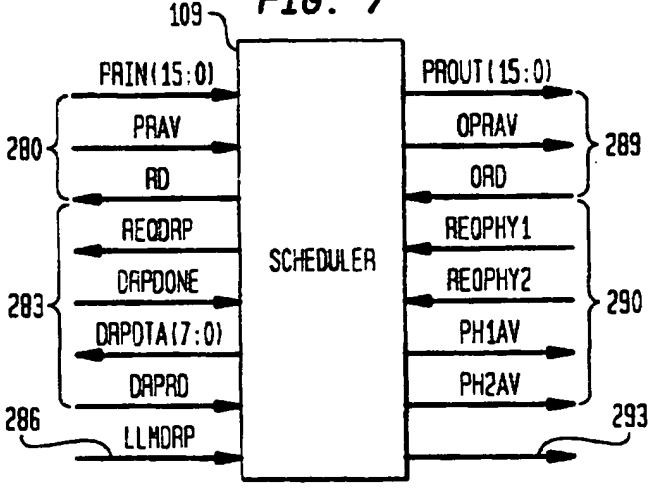


FIG. 8

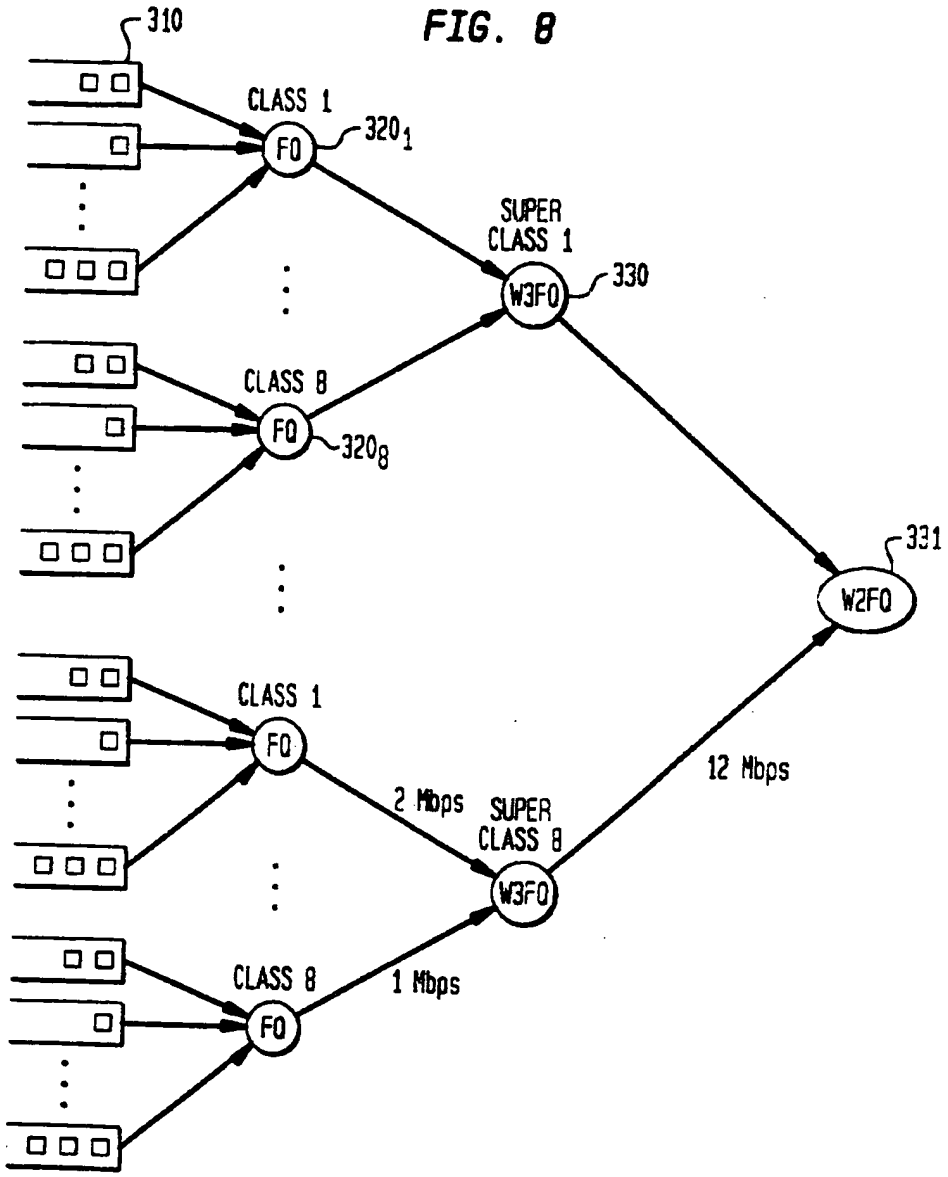

310
CLASS 1
FQ
320_1
SUPER CLASS 1
W3FQ
330
CLASS 8
FQ
320_8
331
W2FQ
CLASS 1
FQ
2 Mbps
SUPER CLASS 8
W3FQ
12 Mbps
CLASS 8
FQ
1 Mbps

FIG. 9A

```
Check if packet is multicast
Store packet in packet record descriptor defined by PageList
For each PHY that packet must be multicasted do
   Add packet to queue Q[I];
   Increase the size of queue I;
   if (Packet DOES NOT HAVE OPTIONS) then
      Increase the size of the queue associated with give PHY
      Buffer management module should update statistics
      if sizeof Q[I] >= 1 then
         if packet is the first for SuperClass SC then
               insert packet in main scheduler for SuperClass SC
         elsif packet is the first for Class C of SuperClass SC then
               insert packet in class scheduler SC for class C
         else
               insert packet to calender queue corresponding to class C and SuperClass SC
      end if;
   else
      Insert Complete packet record in a linked list of packets with Options
   end if;
 end for;
```

FIG. 9B

```
Update global scheduler system potential;
Search the global scheduler table for the superclass with the minimum FP;
Dispatch packet for transmission;
Update queue size of corresponding PHY.
If packet is multicast to multiple Virtual Interfaces then
   Dispatch packet for transmission multiple times, once for
   each Virtual Interface
endif;
Search the SuperClass table for the next packet eligible packet;
if (no more packets in super class) then
     exit;
end if;
Calculate new Starting and Finish potential of super class;
Add entry to global scheduler;
Find the next packet for transmission within class
if (no more packets within class) then
    exit
end if;
Calculate new Starting and Finish potential of class;
Add entry to SuperClass Scheduler;
Select next packet for transmission from the class calendar queue;
add entry in the class scheduler;
```

FIG. 10

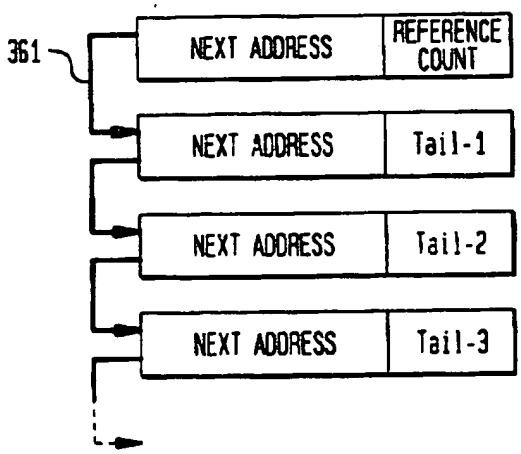


FIG. 11

```
If multicast packet then
        get multicast bitmap
        determine interface processors for forwarding packet
        forward packet record to all these interface processor
        use pointer to head of packet to access reference count and
        update it with the number of interfaces that packet must be
forwarded.
Else
        use PollerIndex to determine outgoing interface processor
        Forward packet to interface processor
end if
Run buffer sizing algorithm
```

FIG. 14

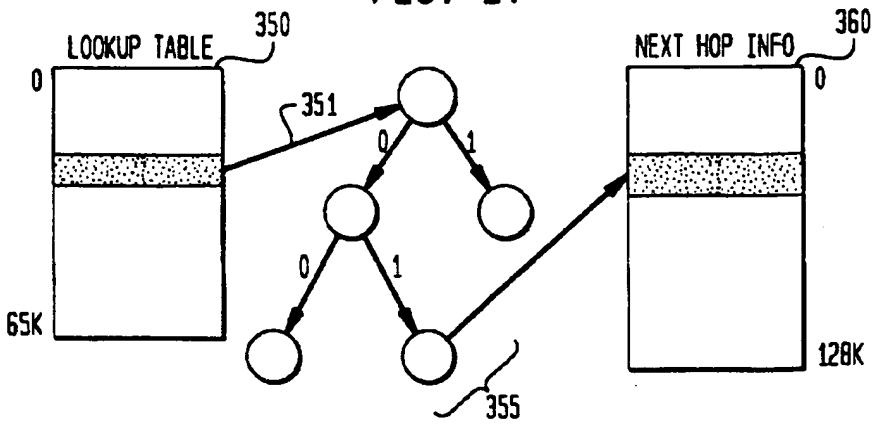


FIG. 12A

```
INTP = INTERFACE PROCESSOR
On packet record arrival from the Flow ID.

If unicast packet then
        Forward packet to INTP[PollerIndex]
        BufferCounter[Pollerindex] += PacketRecord[Size]
        UnicastBuffers += PacketRecord[Size]
else if multicast packet then
        McastBufferCounter += PacketRecord[Size]
        If McastBufferCounter > McastThreshold then
                McastBufferCounter -= PacketSize
                drop packet record
        else
                Forward packet based on multicast bitmap to multiple pollers
        end if
end if
Run Buffer Control Algorithm
```

FIG. 12B

```
INTP = INTERFACE PROCESSOR
On packet deletion

if unicast packet then
        BufferCounter[INTP] -= PacketSize
        UnicastBuffers -= PacketSize
        OutStandingBuffers[INTP] -= PacketSize
        TotalOutstanding -= PacketSize
else if multicast packet then
        McastBufferCounter -= PacketSize
end if
```

FIG. 13

```
Buffer Size Control

    TotalBuffers = McastBufferCounter + UnicastBuffers
    Overflow = MAXBUFFERS - TotalBuffers + TotalOutstanding
    if Overflow the
        select poller such that
        BufferCounter[Poller] - Outstanding[Poller] > MAXBUF[Poller]
        Send delete signals to that poller for a total of
        min(Overflow,MAXBUFF[Poller]) packets;
        Outstanding[Poller] += min(Overflow, MAXBUFF[Poller]);
        TotalOutstanding += min(Overflow, MAXBUFF[Poller]);
    end if
```

FIG. 15A

```
1.  if(PacketRecord[Control]&(IcmpError|DropPacket|PassPacket|RcPacket) then
2.     goto FORWARD
3.  end if;
4.  using 16 most significant bits for hashing
5.  Search corresponding tree
6.  Match corresponding router entry
7.  if (RouterEntry[Flags] & RcAddress) then
8.      PacketRecord[control]| = RcPacket;
9.  endif;
10. if (PacketRecord -> Protocol = IPPROTO_IGMP) |(PacketRecord->DestinationAddress>= 0xe0000000)&&
11.      (PacketRecord[DestinationAddress] <= 0xe00000ff)) then
12.      PacketRecord[Control] |= RcPacket;
13.      goto FORWARD
14. end if;
15. if (No TreeNode or No Route Entry) then
16.     if (PacketRecord[Control] && SsrrIpOption)
17.             PacketRecord[ICMP_CODE] = ICMP_UNREACHABLE_SOURCE_FAIL
18.     else
19.             PacketRecord[ICMP_CODE] = ICMP_UNREACHABLE_NET;
20.     end if;
21.     PacketRecord[ICMP_TYPE] = ICMP_UNREACHABLE
22.     PacketRecord[Control] |= ICMP_ERROR
23.     goto FORWARD
24. end if;
25. if (PacketRecord[Control] & UpdateArpEntry) then
26.     If (RouteEntry exists) then
27.             PacketRecord[MacAddress] = RouterEntry
28.     else
29.             PacketRecord[Control] |= MacInArpCache
30.     end if;
31.     goto FORWARD
32. end if;
33. if ((PacketRecord[Control] & SsrrIpOption)&&(RouteEntry[Flags]&Point2Point)
34.      PacketRecord[IcmpType] = ICMP_UNREACHABLE
35.      PacketRecord[IcmpCode] = ICMP_UNREACHABLE_SOURCE_FAIL
36.      PacketRecord[Control] |= ICMP_ERROR|DROP_PACKET;
37.      goto FORWARD
38. end if
39. if ((RouteEntry[PollerIndex] = PacketRecord[PollerIndex])&&
40.      (RouteEntry[PhyIndex] = PacketRecord[PhyIndex]&&
41.      (PacketRecord[DestinationAddress]>0)          &&
42.     (PacketRecord[SourceAddress]&TreeNode[SubnetMask] = RouteEntry[NextHop])) then
43.             PacketRecord[RouterEntry] = RouteEntry;
44.             PacketRecord[ICMPTYPE] = ICMP_REDIRECT;
45.             PacketRecord[Control] |= ICMP_ERROR|DROP_PACKET;
46.             goto FORWARD
47. end if;
```

FIG. 15B

```
49. PacketRecord[PHYIndex]=RouteEntry[PHYIndex];
50. PacketRecord[PollerIndex]=RouteEntry[PollerIndex];
51. if(RouteEntry[flags]&Gateway) then
52.     PackardRecord[Control]] |= MacInArpCache;
53.     goto FORWARD
54. end if
55. if (NOT(RouteEntry[Flags]&ValidMacFlag)) then
56.      PacketRecord[Control]]|=NeedArpPacket;
57.      PacketRecord[MACaddress]=RouteEntry
58.      goto FORWARD;
59. end if;
60. PacketRecord[MacAddress]=RouteEntry[MacAddress];
61. FORWARD;
62.    forward packet to Flow Identification
```

FIG. 16

```
1.  If (ClassifyFlag = 1) then
2.      Hash_Index = Hash(Source Address, Destination Address, Souce Port, Destination Port, Prot);
3.      FlowRecord = Hash_Table(Hash_Index);
4.      if (FlowRecord = NULL) then
5.            FlowRecord=random(1,64K);
6.      end if;
7.      If (fields in flow record match with fields in packet record) then
8.              update QoS parameters of packet record
9.              update statistics in flow record
10.     else
11.             Clear statistics of flow record.
12.             Replace fields of flow record from fields in packet record
13.             Update Qos parameters of packet record.
14.     End if;
15. end if;
16. if (MacInArpCache = 1) then
17.     MacIndex = DestinationAddress(15 downto 0);
18.     MacTag = MacTable(MacIndex);
19.     if (MacTag = DestinationAddress(31 downto 16)) then
20.           Update MAC address in packet record
21.     else
22.           Set NeedArp flag
23.     end if;
24. end if;
25. update packet record and transmit it
```

FIG. 17A

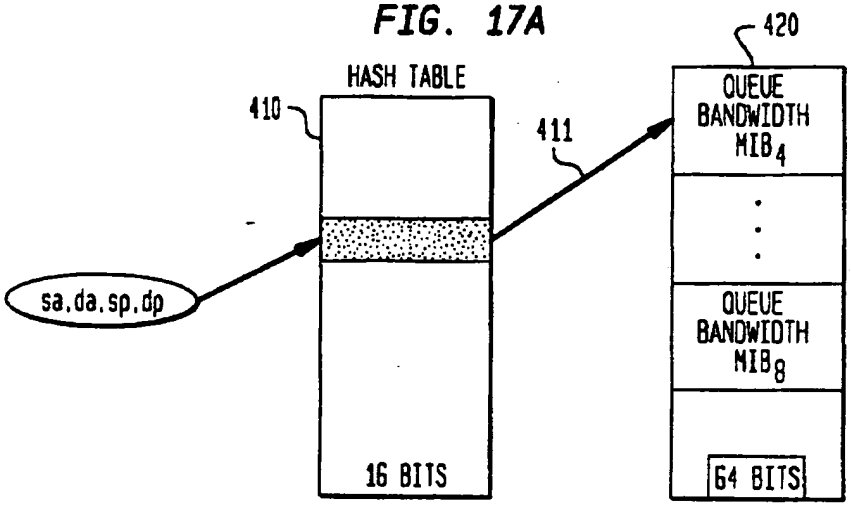


FIG. 17B

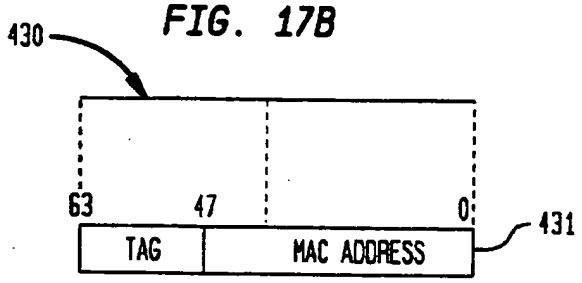


FIG. 18

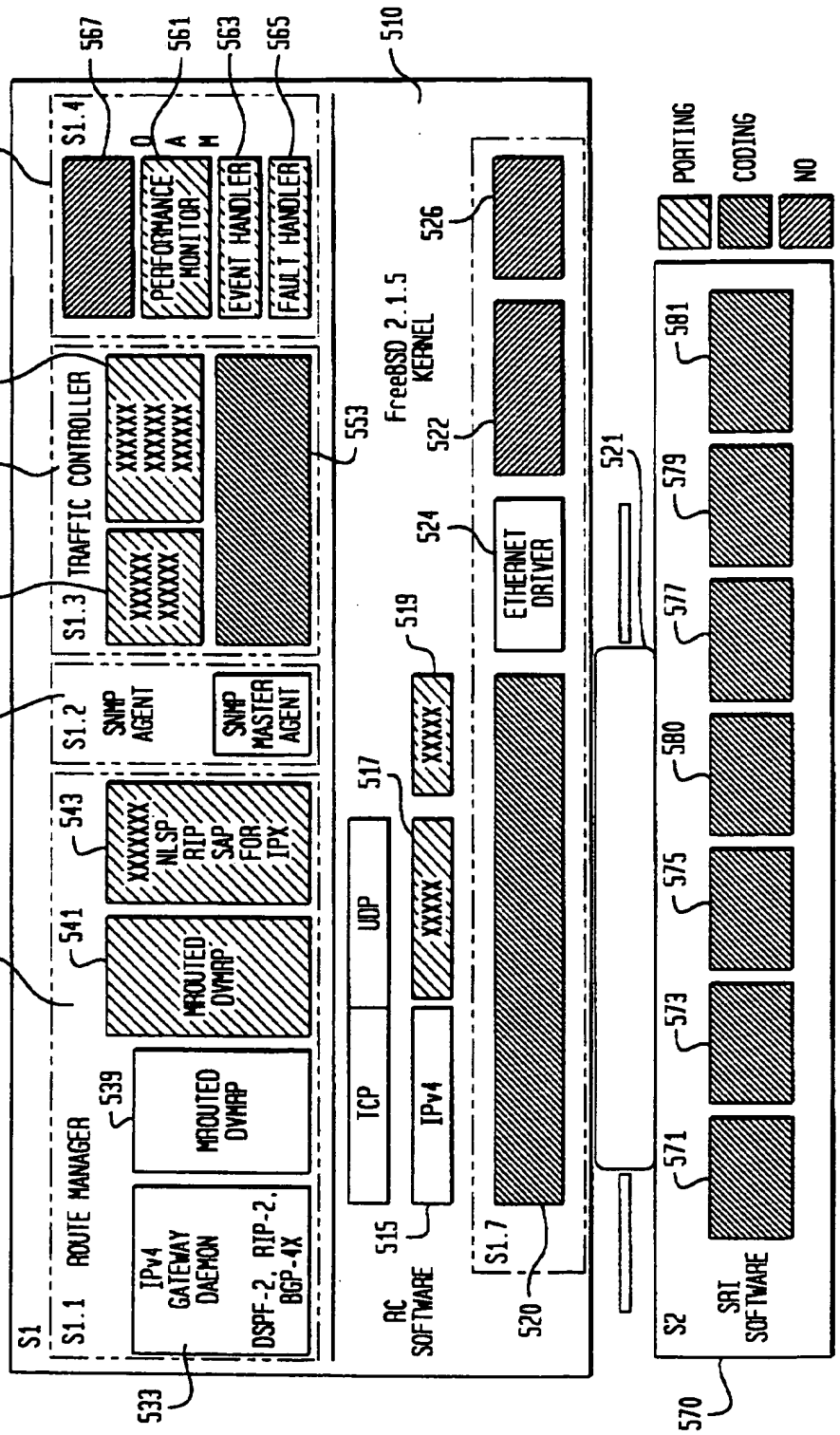

S1
S1.1 ROUTE MANAGER
IPv4 GATEWAY DAEMON
DSPF-2, RIP-2, BGP-4X
MROUTED DVMRP
XXXXXXX NLSP RIP SAP FOR IPX
MROUTED DVMRP
S1.2 SNMP AGENT
SNMP MASTER AGENT
S1.3 TRAFFIC CONTROLLER
XXXXXX XXXXXX XXXXXX
XXXXXX XXXXXX
S1.4
PERFORMANCE MONITOR
EVENT HANDLER
FAULT HANDLER
TCP
UDP
IPv4
XXXXX
XXXXX
RC SOFTWARE
S1.7
FreeBSD 2.1.5 KERNEL
ETHERNET DRIVER
S2
SRI SOFTWARE
PORTING
CODING
NO
510
515
517
519
520
521
522
524
526
530
533
539
541
543
545
551
553
555
557
560
561
563
565
567
570
571
573
575
577
579
580
581

FIG. 19

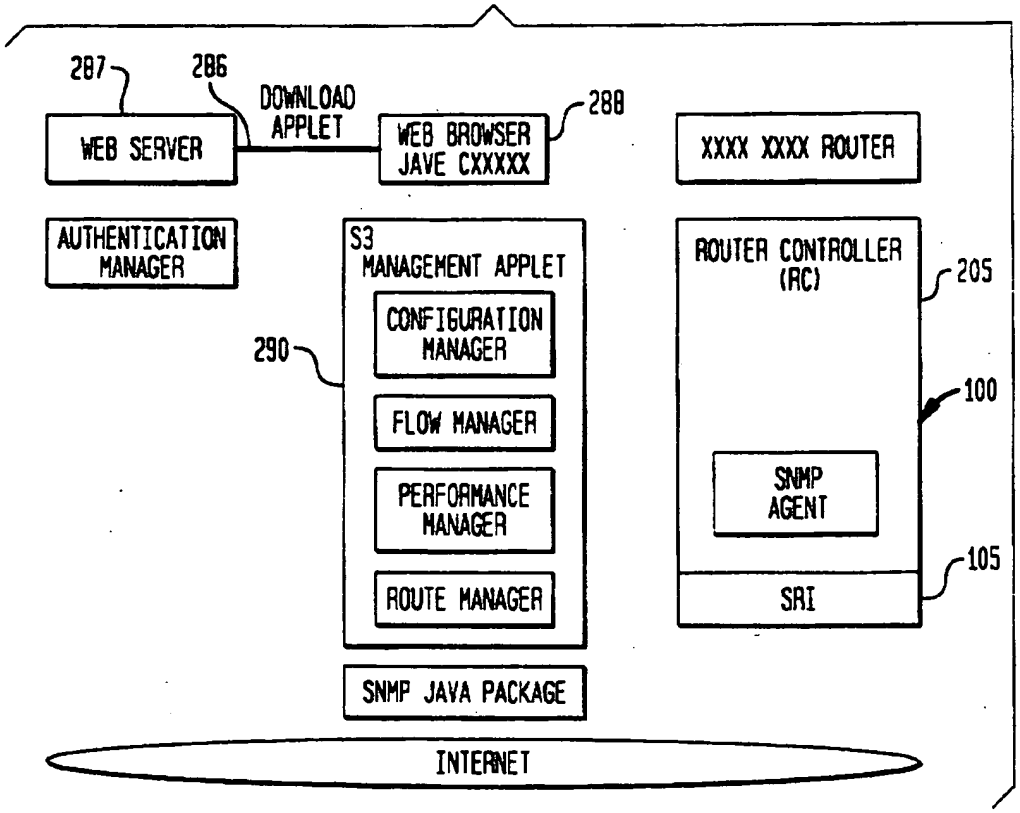
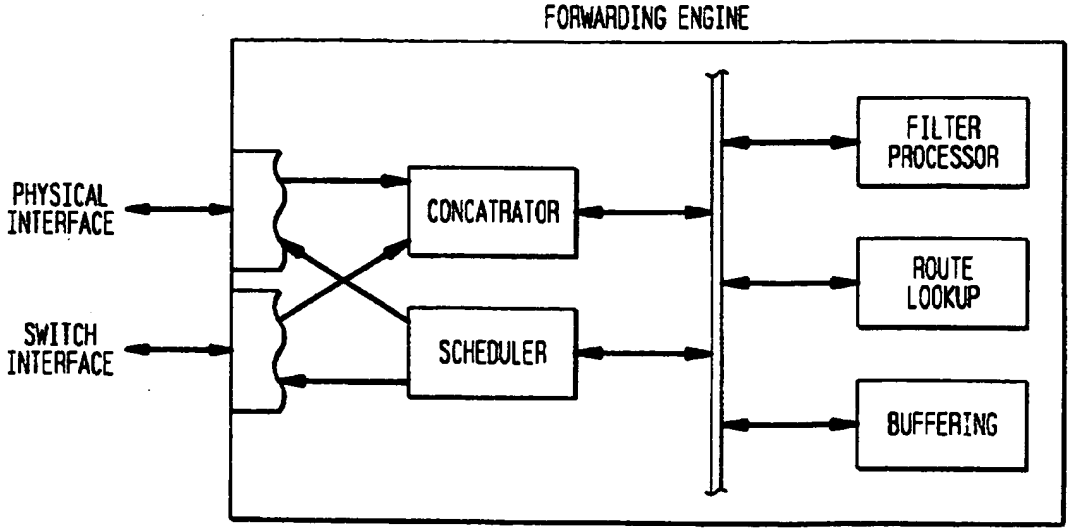

287
286
DOWNLOAD APPLET
288
WEB SERVER
WEB BROWSER JAVE CXXXXX
XXXX XXXX ROUTER
AUTHENTICATION MANAGER
S3 MANAGEMENT APPLET
ROUTER CONTROLLER (RC)
205
290
CONFIGURATION MANAGER
100
FLOW MANAGER
SNMP AGENT
PERFORMANCE MANAGER
105
ROUTE MANAGER
SRI
SNMP JAVA PACKAGE
INTERNET

FIG. 20


FORWARDING ENGINE
PHYSICAL INTERFACE
SWITCH INTERFACE
CONCATRATOR
SCHEDULER
FILTER PROCESSOR
ROUTE LOOKUP
BUFFERING